Revamping the QEMU Memory API

Avi Kivity
Red Hat
Nov 8, 2012

Agenda

- Motivation
- The Old API
- Reality
- Hierarchical memory
- Multiple masters
- Internal data structures

Motivation

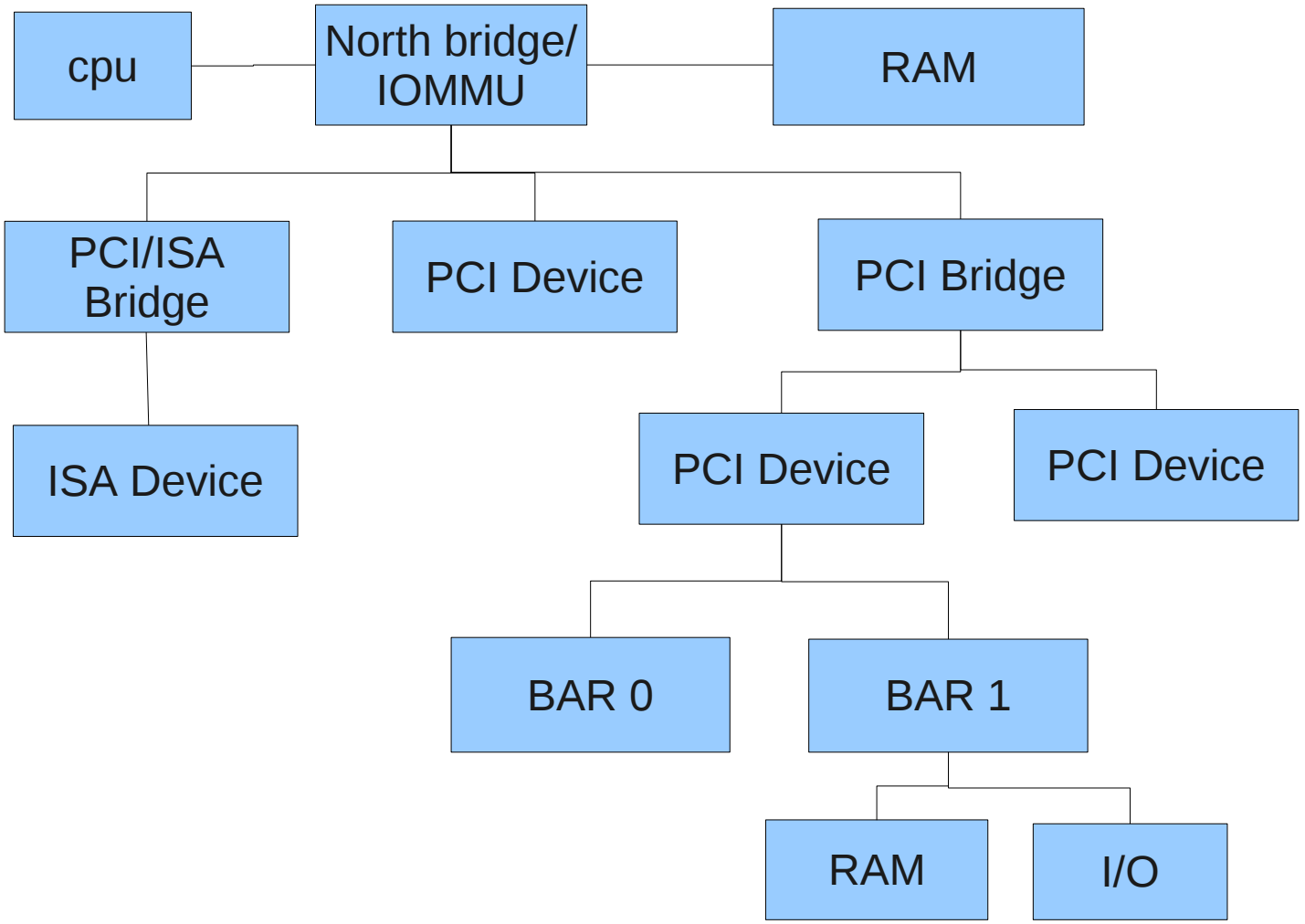
- Memory consumption
- Correctness
- Performance - concurrency
- Features
  - Hierarchical buses
  - Multiple address spaces
- Code deduplication

The Old API

- `cpu_register_io_memory()`
- `qemu_ram_alloc()`
- `cpu_register_physical_memory()`

- Page-based
- Destructive in-place updates
- Pointer arithmetic

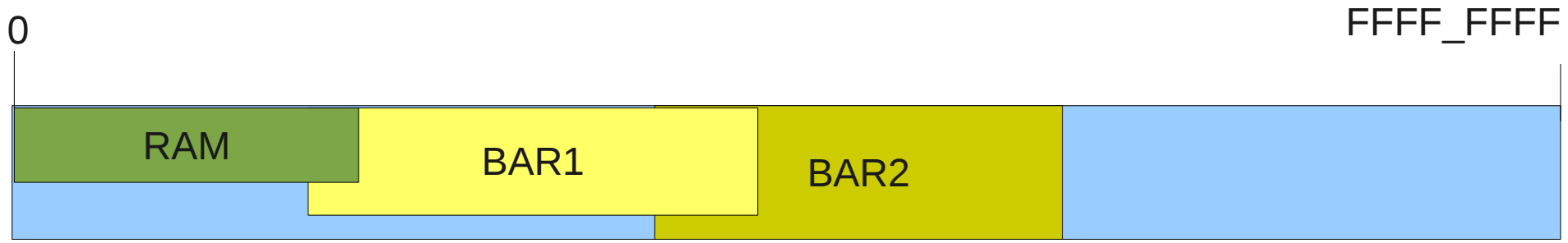
Reality


cpu
North bridge/
IOMMU
RAM
PCI/ISA
Bridge
PCI Device
PCI Bridge
ISA Device
PCI Device
PCI Device
BAR 0
BAR 1
RAM
I/O

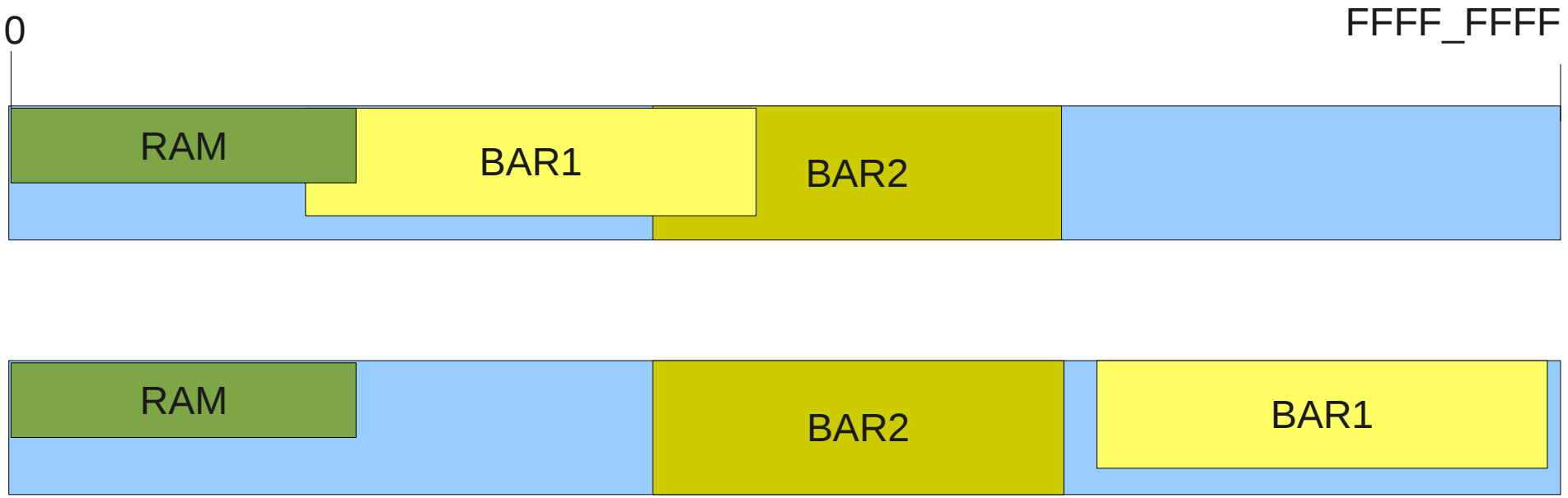
Features of hardware memory routing

- Transactions pass multiple devices until they reach the target
- Intermediate devices can modify addresses, choose among several devices, or terminate the transaction
- Intermediate device configuration can change
- Memory regions can hide one another
- Different initiators see different layouts
- Multiple address space types exist

Example address space change

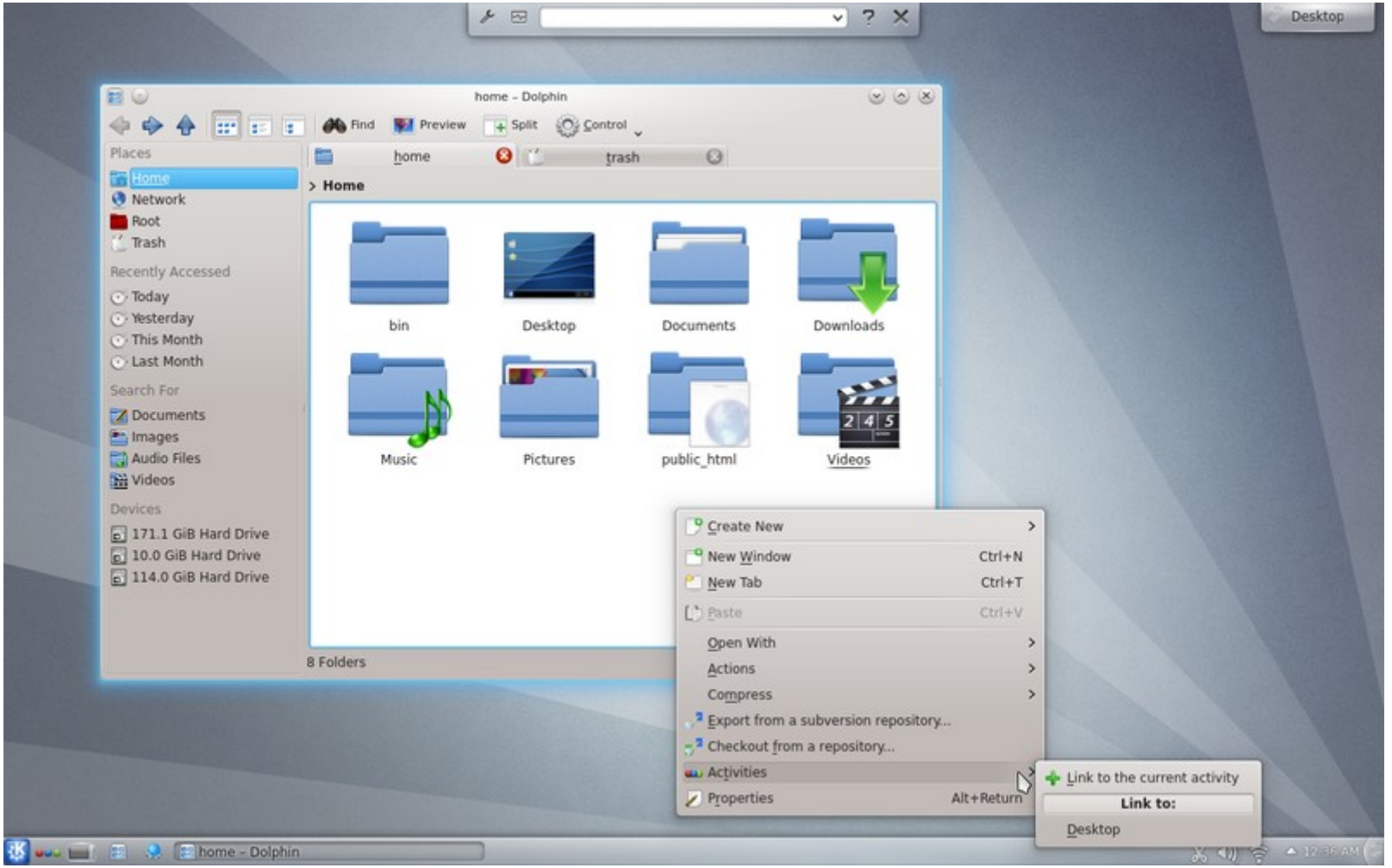


Example address space change



A change in BAR1's base address requires updating parts of both BAR1 and BAR2

An analogy


Desktop
home – Dolphin
Find
Preview
Split
Control
Places
Home
Network
Root
Trash
Recently Accessed
Today
Yesterday
This Month
Last Month
Search For
Documents
Images
Audio Files
Videos
Devices
171.1 GiB Hard Drive
10.0 GiB Hard Drive
114.0 GiB Hard Drive
home
trash
Home
bin
Desktop
Documents
Downloads
Music
Pictures
public_html
Videos
8 Folders
Create New
New Window Ctrl+N
New Tab Ctrl+T
Paste Ctrl+V
Open With
Actions
Compress
Export from a subversion repository...
Checkout from a repository...
Activities
Properties Alt+Return
Link to the current activity
Link to:
Desktop
home – Dolphin

New API

- Hierarchical object model
- Devices only aware of their own regions (e.g. BARs)
- Intermediate devices compose device regions into address spaces
- Memory core responsible for rendering the result

New API

- `memory_region_init*()`
- `memory_region_add_subregion()`
- `memory_region_del_subregion()`

Region types

- I/O
- RAM
- Container
- ROM/Device
- Alias
- IOMMU

Neat features

- Transactions
- Endianness support
- Word size support
- Alignment support
- Mutators

Implementation details

Old implementation:

```
phys_map[addr] = { &object,
                   offset within object }
```

New implementation:

```
phys_map[addr] = &section_x
section_x = { object, offset within object,
              address of section }
```


Variable Depth Radix Tree

- Old implementation used a fixed depth radix tree
  - Element size = 16 bytes
  - 1 element per page
- New implementation uses a variable depth radix tree
  - Similar to x86 page tables
  - 2 byte element size
  - 1 element per page
    - Or 1 element per 1024 pages
      - Or 1 element per 1048576 pages
        - (if they all belong to the same region)

Preparing for RCU

- No in-place updates
- Construct a new structure, replacing the old one

Debuggability

```
(qemu) info mtree
memory
0000000000000000-7fffffffffffffe (prio 0, RW): system
  0000000000000000-0000000007ffffff (prio 0, RW): alias ram-below-4g @pc.ram 000000000000
0000000007ffffff
  00000000000a0000-00000000000bffff (prio 1, RW): alias smram-region @pci 00000000000a00
00000000000bffff
  00000000000c0000-00000000000c3fff (prio 1, R-): alias pam-rom @pc.ram 00000000000c0000
00000000000c3fff
  00000000000c4000-00000000000c7fff (prio 1, R-): alias pam-rom @pc.ram 00000000000c4000
00000000000c7fff
  00000000000c8000-00000000000cbfff (prio 1, R-): alias pam-rom @pc.ram 00000000000c8000
00000000000cbfff
  00000000000ca000-00000000000ccfff (prio 1000, RW): alias kvmvapic-rom @pc.ram 0000000
00000000000ccfff
  00000000000cc000-00000000000cffff (prio 1, R-): alias pam-rom @pc.ram 00000000000cc000-
  00000000000d0000-00000000000d3fff (prio 1, RW): alias pam-ram @pc.ram 00000000000d00
00000000000d3fff
  00000000000d4000-00000000000d7fff (prio 1, RW): alias pam-ram @pc.ram 00000000000d40
00000000000d7fff
  00000000000d8000-00000000000dbfff (prio 1, RW): alias pam-ram @pc.ram 00000000000d80
00000000000dbfff
  00000000000dc000-00000000000dffff (prio 1, RW): alias pam-ram @pc.ram 00000000000dc00
00000000000dffff
```


Memory Listeners

- Callbacks that observe changes to physical address space:
  - `region_add`
  - `region_del`
- Used for anything that needs to know the flattened layout
  - mmio lookup data structure generation
  - kvm, xen
  - vfio, vhost-net

Memory Listeners (cont)

- Additional notifications
  - Dirty logging control
  - Coalesced mmio control
  - ioeventfds

Summary

- An API that matches the world it models
- Easy to use; object-based
- Inclusive
- Accurate; handles corner cases
- Build for performance
- Well documented

Q&A